

Pri tej nalogi sem naredil tabele, za številke sem uporabil INT, za besede pa VARCHAR(100)), ker sem tako videl tudi v začetni tabeli. Tabelam sem nastavil tudi glavni ključ, na določenih tabelah pa še tuj ključ drugih tabel. Da jih bom kasneje lažje združeval. Tabeli pleme sem zapisal podatke z VALUES, ostalim pa sem samo vstavil določeno selekcijo.

**DROP** **TABLE** **IF** **EXISTS** pleme**;**

**CREATE** **TABLE** pleme**(**tid **INT** **PRIMARY** **KEY,** tribe **VARCHAR(**100**));**

**INSERT** **INTO** pleme**(**tid**,** tribe**)** **VALUES** **(**1**,**"Rimljani"**),** **(**2**,**"Tevtoni"**),** **(**3**,**"Galci"**),** **(**4**,**"Narava"**),** **(**5**,**"Natarji"**);**

**DROP** **TABLE** **IF** **EXISTS** aliansa**;**

**CREATE** **TABLE** aliansa**(**aid **INT** **PRIMARY** **KEY,** alliance **VARCHAR(**100**));**

**INSERT** **INTO** aliansa **(**aid**,** alliance**)** **SELECT** **DISTINCT** aid**,** alliance **FROM** x\_world**;**

**DROP** **TABLE** **IF** **EXISTS** igralec**;**

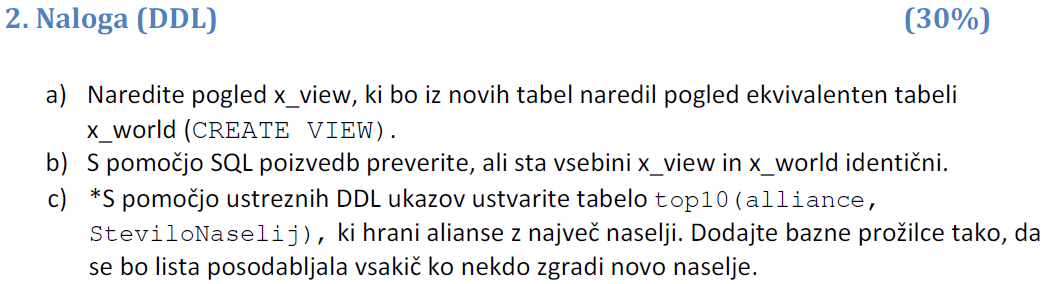
**CREATE** **TABLE** igralec**(**pid **INT** **PRIMARY** **KEY,** player **VARCHAR(**100**),** tid **INT,** aid **INT,** **FOREIGN** **KEY(**tid**)** **REFERENCES** pleme**(**tid**),** **FOREIGN** **KEY(**aid**)** **REFERENCES** aliansa**(**aid**));**

**INSERT** **INTO** igralec **SELECT** **DISTINCT** pid**,** player**,** tid**,** aid **FROM** x\_world **ORDER** **BY** pid**;**

**DROP** **TABLE** **IF** **EXISTS** naselje**;**

**CREATE** **TABLE** naselje**(**id **INT** **PRIMARY** **KEY,** x **INT,** y **INT,** vid **INT,** village **VARCHAR(**100**),** population **INT,** pid **INT,FOREIGN** **KEY(**pid**)** **REFERENCES** igralec**(**pid**));**

**INSERT** **INTO** naselje **SELECT** **DISTINCT** id**,** x **,** y**,** vid **,** village **,** population **,** pid **FROM** x\_world **ORDER** **BY** id**;**



1. **CREATE** **VIEW** x\_view **AS** **SELECT** **\*** **FROM** naselje **NATURAL** **JOIN** aliansa **NATURAL** **JOIN** igralec**;**

* Ustvarjenje pogleda je zelo enostavno, ker sem že pravilno naredil tabele in sem uporabil le NATURAL JOIN.

1. **SELECT** **\*** **FROM** x\_world**;**

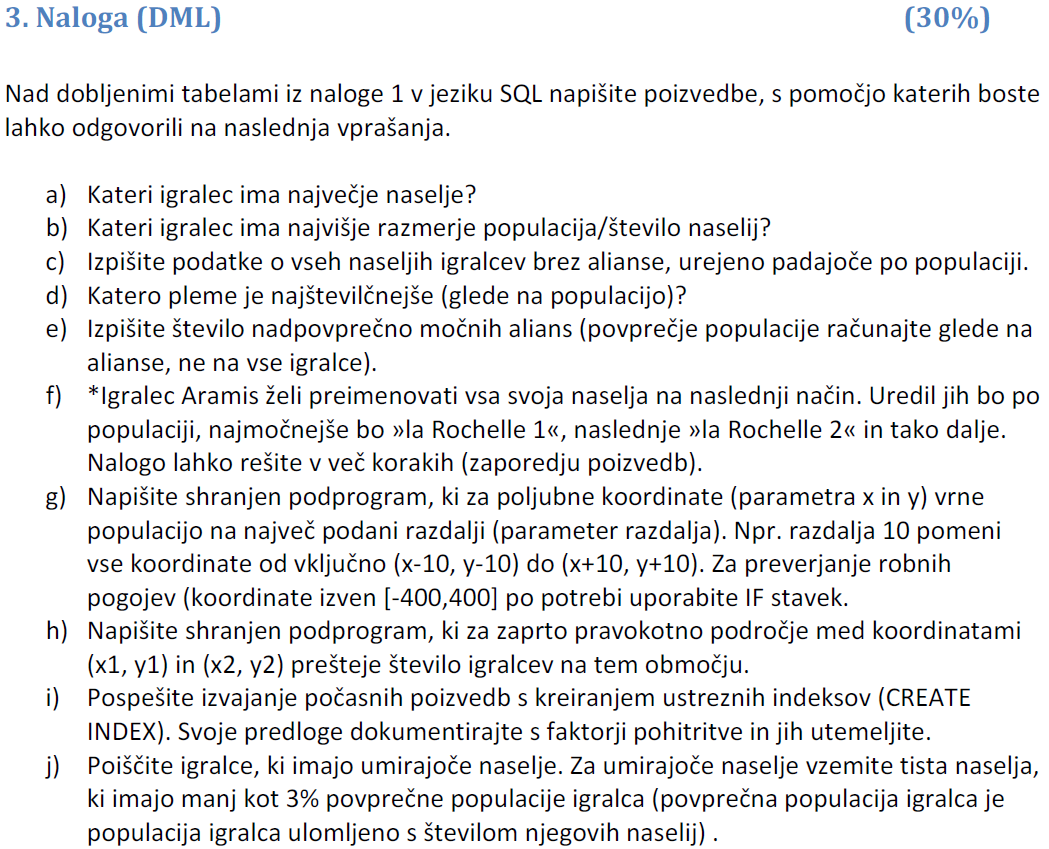
**SELECT** **\*** **FROM** x\_world **WHERE** **NOT** **EXISTS** **(SELECT** **\*** **FROM** x\_view**);**

* Pravilnost tabele sem najprej ročno pregledal, nato pa še progamsko.

1. **CREATE** **TABLE** top10**(**alliance **VARCHAR(**100**)** **PRIMARY** **KEY,** SteviloNaselij **INT);**

**INSERT** **INTO** top10**(**alliance**,** SteviloNaselij**)** **SELECT** allance**,** **COUNT(\*)** **AS** stNaselji **FROM** aliansa **NATURAL** **JOIN** igralec **NATURAL** **JOIN** naselje **WHERE** aid**!=**0 **GROUP** **BY** aid **ORDER** **BY** stNaselji **DESC** **LIMIT** 0**,**10**;**

* Za top10 tabelo sem najprej naredil tabelo in vanjo vstavil selekcijo naselij sortirano po številu naselij z limitom 10. Za prožilec sem pognal le to kodo, vsakič po tem, ko se nekaj vstavi v tabelo. Če bi se to dejansko uporabljalo, bi moral narediti še prožilec za AFTER UPDATE.



1. **SELECT** pid**,** player**,** village**,** population **FROM** naselje **NATURAL** **JOIN** igralec **WHERE** population**=(SELECT** **MAX(**population**)** **FROM** naselje**);**

* Izberemo tam, kjer je populacija največja.

1. **SELECT** player**,** **SUM(**population**)/COUNT(\*)** **AS** ratio **FROM** naselje **NATURAL** **JOIN** igralec **GROUP** **BY** pid **ORDER** **BY** ratio **DESC** **LIMIT** 0**,**1**;**

* Izberemo igralca, kjer je razmerje SUM(population)/COUNT(\*) največje.

1. **SELECT** id**,** vid**,** x**,** y**,** village**,** population **FROM** igralec **NATURAL** **JOIN** naselje **WHERE** aid**=**0 **ORDER** **BY** population **DESC;**

* Združimo naselje in igralec, sortiramo ju po populaciji, kjer je aid enak 0.

1. **SELECT** tid**,** tribe **FROM** igralec **NATURAL** **JOIN** naselje **NATURAL** **JOIN** pleme **GROUP** **BY** tid **ORDER** **BY** **SUM(**population**)** **DESC** **LIMIT** 0**,**1**;**

* Združimo tabeli naselje in pleme, grupiramo vsa plemena, sortiramo jih po SKUPNI populaciji in izpišemo pleme in tid.

1. **SELECT** **COUNT(\*)** **AS** nadpovprecnoMocnih **FROM(SELECT** **sum(**population**)** **AS** p **FROM** igralec **NATURAL** **JOIN** aliansa **NATURAL** **JOIN** naselje **GROUP** **BY** aid**)** **AS** s **WHERE** s**.**p**>(SELECT** **AVG(**s**.**p**)** **FROM** **(SELECT** **SUM** **(**population**)** **as** p **FROM** igralec **NATURAL** **JOIN** aliansa **NATURAL** **JOIN** naselje **GROUP** **BY** aid**)** **as** s**);**

* Združimo tabela igralec aliansa naselje, ker ne morem neposredno združiti naselja in alianse in ju grupiramo po aid-ju in shranimo v *s*, skupno populacijo pa v *p*. Izračunamo povprečje, izberemo vse in jih preštejemo.

1. **SET** @zapSt**:=**0**;**

**UPDATE** naselje **SET** village **=** **concat(**'la Rochelle '**,CAST(**@zapSt**:=**@zapSt**+**1 **AS** **CHAR))** **WHERE** pid**=(SELECT** pid **FROM** igralec **WHERE** player**=**'Aramis'**)** **ORDER** **BY** population **DESC;**

* Nastavimo spremenjivko @zapSt. Posodobimo tabelo naselje z sestavljeno bededo iz @zapSt, ki jo castamo v besedo. Da sem preveril , če je bil zapis pravilen, sem uporabil ukaz z regularnim izrazom: **SELECT** population**,** village**,** pid **FROM** naselje **WHERE** village REGEXP 'la Rochelle \*' **ORDER** **BY** population **DESC;**

1. DELIMITER **|**

**CREATE** **FUNCTION** stIgralcev **(**i **INT,** j **INT)** **RETURNS** **INT**

**BEGIN**

**DECLARE** vrni **INT;**

**IF** **NOT(**i**>**400 **OR** j**>**400 **OR** i**<-**400 **OR** j**<-**400**)** **THEN**

**SELECT** **SUM(**population**)** **INTO** vrni **FROM** naselje **WHERE** **(**x **BETWEEN** i **AND** i**+**9**)** **AND** **(**y **BETWEEN** j **AND** j**+**9**);**

**END** **IF;**

**RETURN** vrni**;**

**END;** **|**

DELIMITER **;**

* Funkcija sprejme dva parametra , IF preveri, če je na igralnem polju, SELECT seštevek populacije naredi na naših koordinatah, rezultat pa zapiše v spremenljivko *vrni* in jo vrne. Uporabiti sem moral tudi drug DELIMITER, da lahko funkcijo končam z END; |. Primer uporabe bi bil SELECT »**SELECT** stIGralcev**(**0**,**100**);**«

1. DELIMITER **|**

**CREATE** **FUNCTION** stIgralcevKvadrat **(**x1 **INT,** y1 **INT,** x2 **INT,** y2 **INT)** **RETURNS** **INT**

**BEGIN**

**DECLARE** vrni **INT;**

**IF** **NOT(**x1**>**400 **OR** x1**>**400 **OR** y1**<-**400 **OR** y1**<-**400 **OR** x2**>**400 **OR** x2**>**400 **OR** y2**<-**400 **OR** y2**<-**400**)** **THEN**

**SELECT** **COUNT(\*)** **INTO** vrni **FROM** **(SELECT** **SUM(**population**)** **FROM** naselje **WHERE** **(**x **BETWEEN** x1 **AND** x2**)** **AND** **(**y **BETWEEN** y1 **AND** y2**))** **AS** x**;**

**END** **IF;**

**RETURN** vrni**;**

**END;** **|**

DELIMITER **;**

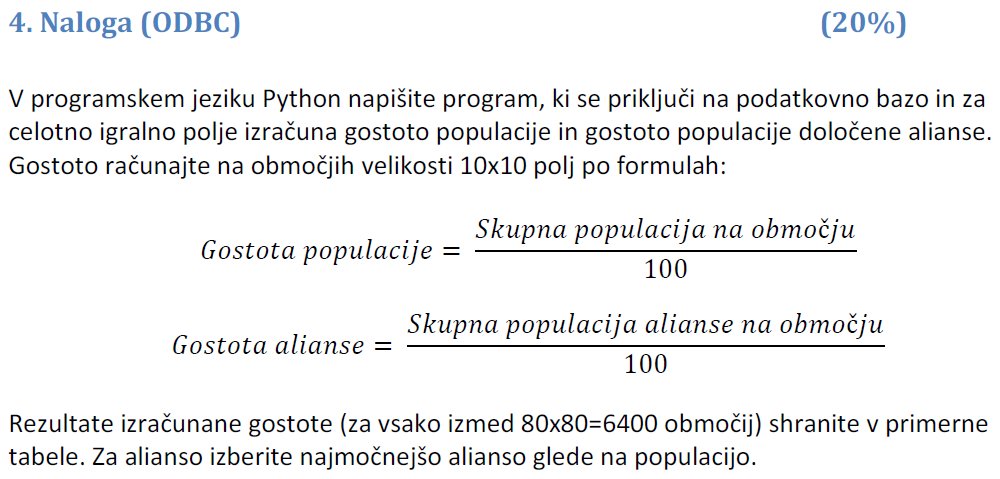
* Ta funkcija vzame 4 parametre, IF preveri, če so v našem območju. Če so, potem naredi SELECT v tem območju med x1, y1 in y1, y2. Select prešteje, koliko rezultatov imamo na tem območju. Primer uporabe : »**SELECT** stIGralcevKvadrat**(**0**,**0**,**100**,**100**);**«

1. **CREATE** **INDEX** naselja **ON** naselje**(**x**,**y**,**population**)** **USING** HASH**;**

* Odločil sem se, da naredim index na tabeli naselje(x, y, population). Ko sem naredil program v Pythonu, je izvajanje trajalo 1:23 sekund, po ustvarjanju tega indexa sem čas izvajanja zmanjšal na 5 sekund, kar je 16,2 krat hitreje.

1. **SELECT** **\*** **FROM** **(SELECT** pid**,** player**,** **SUM(**population**)/COUNT(\*)** **AS** ratio **FROM** naselje **NATURAL** **JOIN** igralec **GROUP** **BY** pid**)AS** x **WHERE** x**.**ratio**<=**3**;**

* Izračunamo povprečje igralca in vseh njegovih naselij v tabelo in pogledamo, kje je razmerje manjše ali enako 3.



Za uspešno priključitev na podatkovno bazo v programskem jeziku Python 3.2 sem moral namestiti modul pyodbc.

Na začetku programa imam nekaj nastavitev za pyqt4 widget, na katerega rišem graf in gradient barvo za označitev naselij. S tem modulom naredimo povezavo z pyodbc.connect() in na tem objektu odpremo cursor(), na katerega lahko pošiljamo ukaze in sprejemamo odgovore MySQL strežnika. Najprej moram zbrisati tabelo, če obstaja, in narediti novo, da ne bi prišlo do napak. Nato grem s for zanko od -400 do 400 in s korakom po 10 za x in nato po y. Sedaj lahko izvajam funkcijo, ki smo jo napisali v 3. nalogi stIgralcev() in ji podam x in y. Funkcija mi vrne gostoto populacije na tem območju, če seveda sploh je kdo na tem območju. Če ni nobenega, pa samo preskočim na naslednjo lokacijo. Ker zdaj vem, da na tem območju obstaja naselje, lahko še naredim poizvedbo najmočnejše alianse tega območja, ki ima največ populacije v vseh naseljih tega območja. Te podatke zapišem v tabelo gostot, ki jo bom uporabil v Excelu in narišem krog na poziciji x, y, velikosti populacija/700, da dobim nekaj primerne velikosti. Na koncu še pustim izvajanje programa in varno zaprem vse povezave. Za pogled tabele lahko uporabimo:

**SELECT** topAid**,** x**,** y**,** gostota**,** alliance **FROM** gostote g**,** aliansa a **WHERE** g**.**topAid**=**a**.**aid **ORDER** **BY** gostota **DESC;**

Python3.2 koda:

**import** pyodbc

**from** PyQt4**.**QtCore **import** **\***

**from** PyQt4**.**QtGui **import** **\***

app**=**QApplication**([])**

widget **=** QDialog**()**

widget**.**setWindowTitle**(**"Gostota populacije"**)**

widget**.**setLayout**(**QVBoxLayout**())**

widget**.**layout**().**setMargin**(**2**)**

widget**.**scene **=** QGraphicsScene**(**widget**)**

widget**.**scene**.**setBackgroundBrush**(**Qt**.**white**)**

widget**.**view **=** QGraphicsView**(**widget**.**scene**,** widget**)**

widget**.**view**.**setAlignment**(**Qt**.**AlignLeft **|** Qt**.**AlignTop**)**

widget**.**view**.**setSizePolicy**(**QSizePolicy**.**MinimumExpanding**,** QSizePolicy**.**MinimumExpanding**)**

widget**.**view**.**setRenderHints**(**QPainter**.**Antialiasing **|** QPainter**.**SmoothPixmapTransform**)**

widget**.**layout**().**addWidget**(**widget**.**view**)**

widget**.**resize**(**800**,** 800**)**

widget**.**scene**.**setSceneRect**(**0**,** 0**,** widget**.**view**.**width**(),** widget**.**view**.**height**())**

widget**.**view**.**setSceneRect**(**0**,** 0**,** widget**.**view**.**width**(),** widget**.**view**.**height**())**

widget**.**show**()**

widget**.**raise\_**()**

grad **=** QRadialGradient**()**

grad**.**setColorAt**(**0**,**QColor**(**0**,**46**,**180**,**100**))**

grad**.**setColorAt**(**1**,**QColor**(**0**,**46**,**180**,**0**))**

conn **=** pyodbc**.**connect**(**'Driver={MySQL ODBC 5.1 Driver};Server=127.0.0.1;Port=3306;Database=seminarska;User=seminarskadb; Password=seminarska;Option=3;'**)**

cur **=** conn**.**cursor**()**

cur**.**execute**(**"DROP TABLE IF EXISTS gostote;"**)**

cur**.**execute**(**"CREATE TABLE gostote(topAid INT REFERENCES aliansa(aid), x INT, y INT, gostota FLOAT);"**)**

**for** x **in** range**(-**400**,**410**,**10**):**

**for** y **in** range**(-**400**,**410**,**10**):**

cur**.**execute**(**"SELECT stIgralcev("**+**str**(**x**)+**","**+**str**(**y**)+**");"**)**

p **=** cur**.**fetchone**()[**0**]**

**if** p**:**

cur**.**execute**(**"SELECT aid, SUM(population)/100 FROM naselje NATURAL JOIN igralec NATURAL JOIN aliansa WHERE (x BETWEEN "**+**str**(**x**)+**" AND "**+**str**(**x**+**9**)+**") AND (y BETWEEN "**+**str**(**y**)+**" AND "**+**str**(**y**+**9**)+**") GROUP BY aid ORDER BY SUM(population) DESC LIMIT 0,1;"**)**

cur**.**execute**(**"INSERT INTO gostote(gostota,x,y,topAid) VALUES ("**+**str**(**p**/**100**)+**", "**+**str**(**x**)+**", "**+**str**(**y**)+**", "**+**str**(**cur**.**fetchone**()[**0**])+**");"**)**

p **/=** 700

grad**.**setRadius**(**p**)**

elipsa **=** widget**.**scene**.**addEllipse**(-**p**,** **-**p**,** 2**\***p**,** 2**\***p**,** QPen**(**QBrush**(**Qt**.**transparent**),** 1**),**QBrush**(**grad**))**

elipsa**.**setPos**(**x**+**400.5**,** y**+**400.5**)**

widget**.**scene**.**update**()**

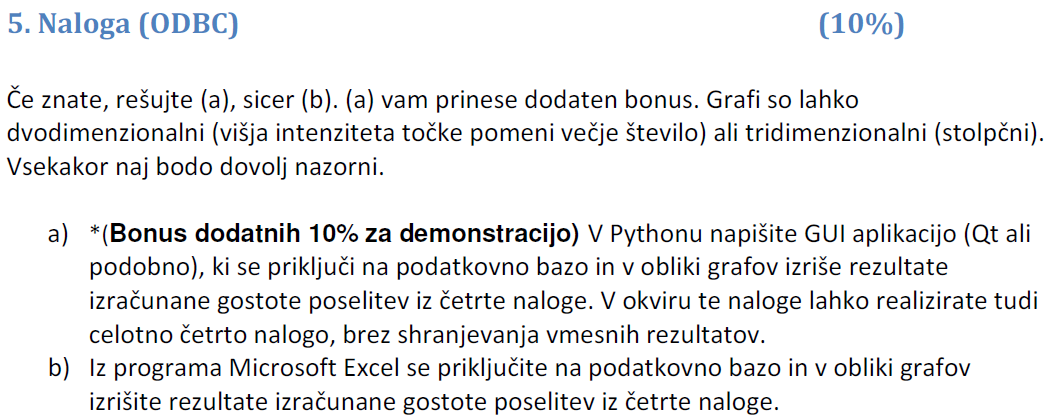
qApp**.**processEvents**()**

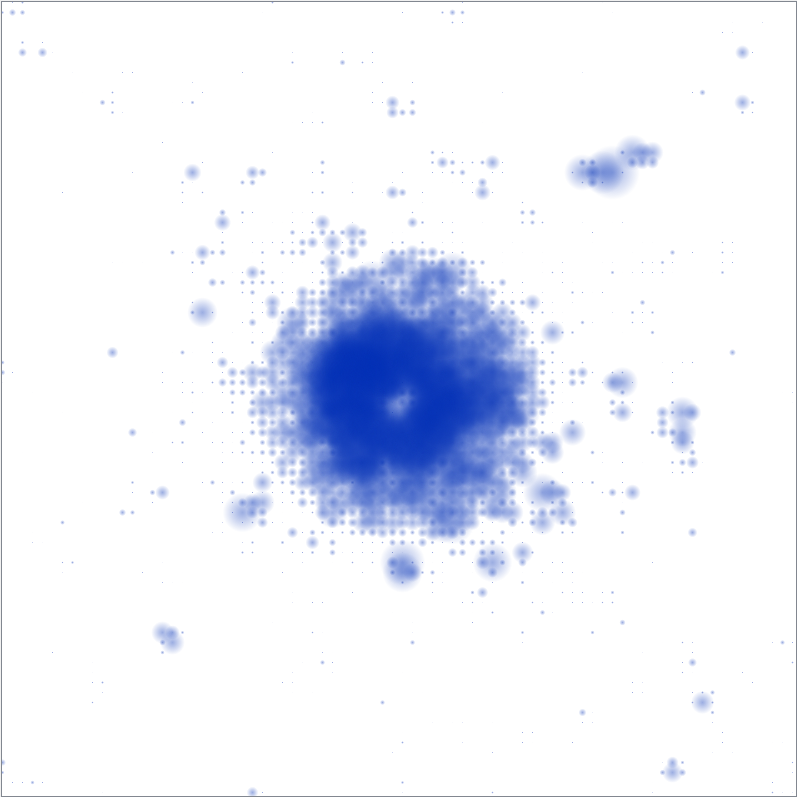
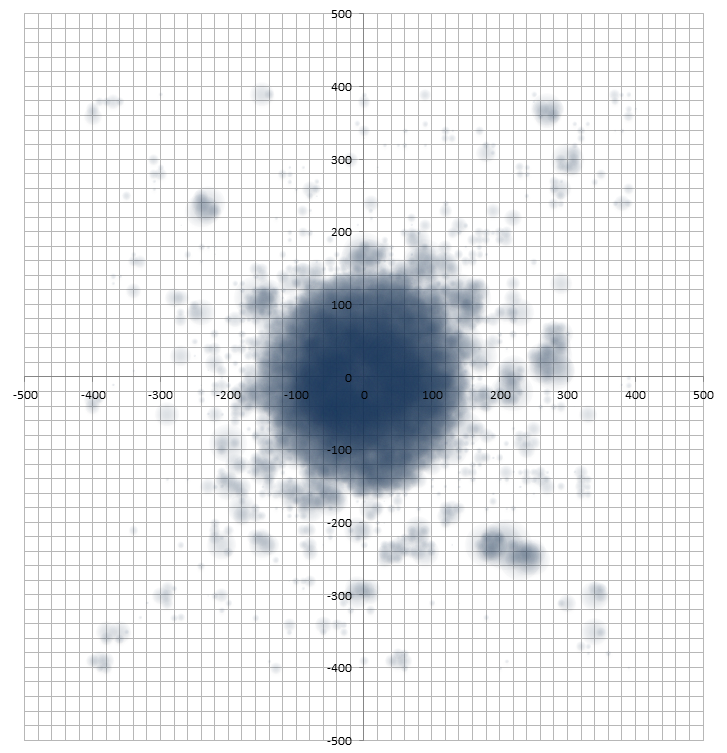
cur**.**close**()**

conn**.**commit**()**

conn**.**close**()**

app**.**exec\_**()**



1. Program je narejen samo z PyQt-jem in samo riše kroge na lokacijah nastavi radij na število populacije in nastavi gradiant barvo, ki je na sredini najtemnejša, ker se pa krogi sekajo pa barva nastane temnejša na gostejše naseljenih območjh.
2. Če hočemo Excel priključiti na MySQL bazo, moramo dodati ODBC data source za MySQL in vnesti naše podatke za povezavo. Potem lahko z Excelom uvozimo tabelo in iz njega izrišemo graf.